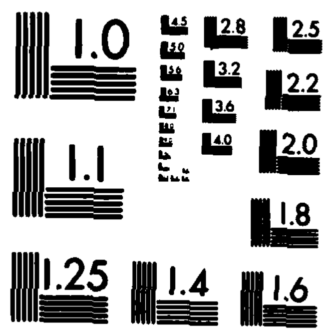
AD-A146 627 TIME SERIES ANALYSIS PROGRAMS FOR STRATIGRAPHIC DATA (U) SCRIPPS INSTITUTION OF OCEANOGRAPHY LA JOLLA CA P SCHIFFELBEIN JUL 84 SIO-REF-84-22 N00014-80-C-0440 1/1

UNCLASSIFIED F/G 9/2 NL




'OPY RESOLUTION TEST CHART

# TIME SERIES ANALYSIS PROGRAMS FOR STRATIGRAPHIC DATA

## SIO REFERENCE SERIES

Paul Schiffelbein







SIO Reference 84-22

University of California Scripps Institution of Oceanography
July 1984

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

| REPORT DOCUMENTATION PAGE | | READ INSTRUCTIONS BEFORE COMPLETING FORM |
|---|---|---|
| 1. REPORT NUMBER | 2. GOVT ACCESSION NO. AD-A146 | 3. RECIPIENT'S CATALOG NUMBER 27 |
| 4. TITLE (and Subtitle) TIME SERIES ANALYSIS PROGRAMS FOR STRATIGRAPHIC DATA | | 5. TYPE OF REPORT & PERIOD COVERED |
| | | 6. PERFORMING ORG. REPORT NUMBER Ref 84-22 |
| 7. AUTHOR(s) Paul Schiffelbein | | 8. CONTRACT OR GRANT NUMBER(s) N00014-80-C-0440 |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS Scripps Institution of Oceanography La Jolla, California 92093 | | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS |
| 11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Arlington, Virginia 22217 | | 12. REPORT DATE July 1984 |
| | | 13. NUMBER OF PAGES |
| 14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) | | 15. SECURITY CLASS. (of this report) Unclassified |
| | | 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE |

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release. Distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report documents a set of FORTRAN programs and subroutines for analyzing stratigraphic time series data. Each project is briefly explained to provide a context for the programs. All algorithms used in the project are also explained. All programs were written for the UCSD Vax 11/780 (VMS) computer.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

University of California, San Diego

Scripps Institution of Oceanography

La Jolla, California 92093

# TIME SERIES ANALYSIS PROGRAMS FOR STRATIGRAPHIC DATA

Paul Schiffelbein

This work was supported by the U.S. Office of Naval Research (Contract #N00014-80-C-0440 to W. H. Berger).

July 1984 SIO Ref. 84-22

Acknowledgements

Bill Coulbourn, Alan Chave and Scott Hills have strongly influenced and promoted my progress in computer programming. I greatly appreciate their help. Wolf Berger provided the inspiration for a number of the projects related to these programs. My sincere thanks to Carol Taunton for typing the manuscript.

Contents

# I. Introduction

This report documents a set of FORTRAN programs and subroutines for analyzing stratigraphic time series data. These programs were written as part of the author's Ph.D. dissertation (Schiffelbein, 1984) and can be considered as an appendix to that work. This documentation is divided into two sections. The first section follows the organization of the author's thesis. Each project is briefly explained to provide a context for the programs. All algorithms used in the project are also explained. The second section provides an alphabetical listing of the algorithms referred to in the first section. All programs were written for the UCSD Vax 11/780 (VMS) computer. Subroutines referred to in the programs but not listed (CARTESIAN, CLEAR, COMPLEX, CONVOLVE, FFT, FFTINVERSE, INTERPOLATE, POLAR, POWER, POWERDB, PROLATE, REALPART, ZEROMEAN) are from a library kindly made available to the author by Alan Chave and are described and listed in Chave (1980).

# II. Project and Program Descriptions

## A. The Effect of Benthic Mixing on the Information Content of Deep-Sea Stratigraphic Signals

Abstract (Chapter 3 of the dissertation):

Benthic mixing or bioturbation affects sediments in a number of ways, including 1) the production of trace fossils, 2) mechanical and/or chemical alteration of the sediment, and 3) filtering or smearing stratigraphic signals. Since mixing alters both the slopes and amplitudes of any recorded events, some knowledge of the process is essential for correct interpretation of the signals. Particularly in the light of recent trends towards high-resolution stratigraphy and signal unmixing, the frequency characteristics of the benthic mixing filter must be understood to know which types of signals can be detected after mixing. Ash and tektite profiles in deep-sea cores from various geographic regions are examined in the frequency domain to determine the characteristics of the benthic mixing process. Analysis of these profiles indicates that even signals from cores having sedimentation rates as high as 7 cm/ka will show severe attenuation of frequencies higher than 0.35 cycles/ka (periods shorter than 2.9 ka) resulting in loss of the ability to resolve closely-spaced events. Most signals will experience much more serious high-frequency attenuation, however. The severity of high frequency loss is directly related to sedimentation rate ($R$ = 0.97 for the cores examined), suggesting that this is the most

important variable when considering a core for paleoceanographic or paleoclimatic study.

1. Program PROJECT: This program computes the power spectrum of an impulse response function (IRF). The IRF is stored in some input file. The power spectrum is given in decibels and stored in a user-specified output file.

2. Program GSTEST is a simple driver program for the subroutine GSIRF. The user is asked for the benthic mixing parameters, which are passed to GSIRF, and the mixing function is generated and stored in a user-specified file.

3. Subroutine GSIRF uses the approximation equation of Officer and Lynch (1983) as a solution to the benthic mixing equation of Guinasso and Schink (1975). The subroutine is accessed by a statement:

   CALL GSIRF [generated IRF vector, mixed layer thickness (cm), diffusion coefficient ($cm^2$/ka), sedimentation rate (cm/ka)]. The output is the benthic mixing impulse response function.

## B. Extracting the Benthic Mixing Impulse Response Function: a Constrained Deconvolution Technique

Abstract (Chapter 4 of the dissertation):

Benthic mixing or bioturbation acts as a low-pass filter on stratigraphic signals, altering both the apparent rates and amplitudes of recorded events. The mixing process is represented as a time-invariant convolution, and uses an impulse response function (IRF) parameterized in terms of mixed layer thickness and mixing intensity (diffusion coefficient). Such parameterization facilitates construction of the inverse (deconvolution) model, which is used to examine the effects of incorrect IRF characterization on deconvolution results. Results show that mixing intensity is as important a parameter as mixed layer thickness when unmixing a stratigraphic signal.

Stable oxygen isotope records from two different species of planktonic foraminifers with changing relative abundances in the same core will show apparent offsets in the timing of events due to mixing (Hutson, 1980). The offset can be used, with the appropriate unmixing equation, to constrain mixed layer thickness and mixing intensity based on analysis of actual isotope signals. The assumption is that any offsets in the records of the two species are totally a product of the mixing process.

The technique was applied to a high-resolution glacial-

interglacial record (*Termination II) from an equatorial* Pacific piston core. Multiple lateral subsamples were stacked to increase signal to noise ratio. A multidimensional nonlinear optimization routine (SIMPLEX) was used to minimize an error function related to stratigraphic offsets between the two species. A solution to this problem yielded a mixed layer 5 cm thick and a mixing intensity of 7 $cm^2$/ka. This mixed layer thickness is at the low end of estimates based on $^{14}C$ profiles in box cores from the same geographic area (Berger and Killingley, 1982). The mixing operator is probably conservative, and does not take into account that sedimentation rates were higher during the last glacial period than during the Holocene. This may explain the inability of the algorithm to completely remove the stratigraphic offset between the two isotope records. The unmixed record (using the parameters 5 cm and 7 $cm^2$/ka) shows a deglaciation overshoot which is distinctly smaller than the "meltwater spike" proposed by Berger et al. (1977), i.e., less than 0.1 $^{o}$/oo versus a range of 0.3 $^{o}$/oo to 1 $^{o}$/oo.

1. Program HUTSON: This program performs a benthic mixing (convolution) operation on an input stable isotope signal. The mixing takes into consideration the change in relative abundance of the isotope carrier (foraminifer species) as described in Hutson (1980). The user is asked for input filenames for isotopes, abundances and the mixing operator. The mixed record is stored in a user-specified file.
2. Program DECOTEST: This program is a simple driver for time-domain deconvolution of a stable isotope signal. The user is asked for the isotope record infile, the mixing function infile, and a constant that controls the sensitivity of the unmixing. Values between 0.001 and 0.05 were found appropriate for most real and simulated isotope signals, with larger numbers giving less sensitivity. The program accesses subroutine DECO, which performs the deconvolution. The unmixed record is stored in a user-specified file.
3. Program HUTDEC: This program is the inverse of program HUTSON. Like DECOTEST, this routine deconvolves an isotope signal, but the relative abundance of the signal carrier is also considered in HUTDEC. The program operates as described for DECOTEST above.
4. Program OFFSET: This is a large and relatively complicated driver program that attempts to remove stratigraphic offset between two stable isotope signals from different foraminifer species in a single core by deconvolving the signals with various unmixing functions. (Refer to the thesis for details.) The heart of the driver is a simplex optimization

algorithm (Nelder and Mead, 1965) modified from Daniels (1978). The fit of the two unmixed curves is optimized by adjusting the benthic mixing parameters. The program examines stratigraphic offset by calling subroutine ERROR, which, by calling GSIRF and DECO, unmixes the two signals and examines closeness of fit.

5. Subroutine ERROR: This algorithm calculates a value for the optimization error function as described for OFFSET. This is a specialized subroutine which accepts two isotope signals (different foraminifer species from the same core) their relative abundances, and the benthic mixing parameters. Four calls to DECO unmixes the records and calculates the residual between the two unmixed signals (error function).

6. Subroutine DECO: This algorithm performs a time-domain deconvolution of a measured or theoretical stable isotope signal. The subroutine is accessed by a statement:

   CALL DECO [IRF vector, number of points in input, input vector, output vector, deconvolution sensitivity].

   DECOTEST is a simple driver program for using this subroutine.

7. Subroutine CNVLV: This subroutine convolves two vectors and stores the result in a third. The input arrays have lengths N and M, with the output of length N + M - 1.

8. Subroutine ZERO: This algorithm sets the mean value of a vector to zero. The original mean value is saved as AMEAN.

C. Bruhnes Dissolution Cycle: Effects on Oxygen Isotopes, Sedimentation Rates, and Signal Spectra

Abstract (Chapter 5 of the dissertation):

A long wavelength cycle (400-500 ka) has been found in stable oxygen isotope records from Pleistocene deep-sea sediments in the Pacific Ocean. This cycle shows an increase in amplitude with increasing water depth, correlates in both phase and duration with the Brunhes dissolution cycle of Adelseck (1980) and is apparently a result of differential dissolution. This dissolution cycle has significantly affected sedimentation rates, particularly in deep water sediments. Sediment loss through dissolution can be quantified by comparing isotope stage lengths with some shallow reference core or by using %$CaCO_3$ data. Examination of a number of equatorial Pacific cores suggests that even sediments recovered from a water depth of 3500 m have undergone significant distortion as a result of this dissolution. Long-wavelength sedimentation rate perturbation results in a widening of spectral peaks, with a consequent decrease of spectral resolution, if some depth scale correction is not made.

The Brunhes dissolution cycle is apparently one of many cycles which continue with a periodicity of roughly 400 ka back into the Miocene. Dissolution on this time scale resembles the orbital eccentricity record, which has strong 100 and 400 ka components. The fact that the 400 ka cycle is

not seen in shallow water oxygen isotope records suggests that it is not directly related to ice volume, but rather to some element of the carbon cycle affecting carbonate understaturation in the deep ocean.

1. Program DISSOLVE: This program accepts two stable oxygen isotope stratigraphies and their isotope stage boundary positions, aligns the second signal to the first using the isotope stage boundaries, and subtracts the second curve from the first. Interpolation is done with cubic splines (de Boor, 1978). The residual curve is then smoothed and stored in a user-specified file.

2. Program ANALPOW: This program computes the power spectrum of a stratigraphic signal. A matrix of downcore measurements and depths are read in, equally spaced with cubic spline interpolation, windowed and Fourier transformed. The raw spectrum is band averged (see Otnes and Enochson, 1972) and converted to relative decibels.

## D. Spectral Effects of Time-Depth Nonlinearities in Deep Sea Sediment Records: A Demodulation Technique for Realigning Time and Depth Scales

Abstract (Chapter 7 of the dissertation):

$^{14}C$ dating and $\%CaCO_3$ in late Pleistocene sediments suggest that deep-sea sedimentation rates vary cyclically and that this cyclicity is related to climate. Sedimentation rate variability leads to nonlinearity in the time-depth mapping function. This nonlinearity can have profound effects on signal spectra, leading to the development of harmonics and intermodulation tones. These distortion effects in the spectra give a direct indication of the degree of nonlinearity, thereby providing a tool for realigning time and depth scales. A tuning technique is developed which assumes a direct link between climate (as measured in O-18 from planktonic foraminifer tests) and sedimentation rates. A criterion of "spectral simplicity", as quantified in the varimax norm, is used to demodulate the input spectrum. Application of this technique to an equatorial Pacific piston core (ERDC 84) found peak glacial sedimentation rates to be 30% higher than peak interglacial rates, a figure in good agreement with $^{14}C$-based estimates from the same area. This technique is compatible with other time-scale tuning techniques such as those using orbital parameters, and, in

combination with these other techniques, provides a method for fine-tuning any late Pleistocene record.

1. Program WAVETEST: This program examines the effects of systematic depth-scale perturbation on signal spectra. A sinusoidal signal is generated by calling WAVGEN. The program then adjusts depth values according to signal amplitude, interpolates with cubic splines for equal sample spacing, and calculates the power spectrum. A range of degrees of signal distortion are examined via a DO loop and the results in both time and frequency domains are stored in a user-specified file.
2. Program ISOJUST: This program is essentially the same as WAVETEST except that it uses measured isotope data as input rather than calling for a sinusoidal signal. The input signal is progressively distorted via a DO loop, and the power spectrum is calculated at each step. The spectra are band averaged and a varimax norm is calculated for each spectrum to determine its simplicity. Isotope stage boundaries are read in, which are used to quantify the signal distortion.
3. Program VARIMAX: This simple program calculates a varimax norm as described by Wiggins (1978) for quantifying the simplicity of a signal. The program accepts a vector as input.
4. Subroutine WAVGEN: This algorithm generates a multi-component sinusoidal time series of length NS. The component wavelengths are specified within the subroutine.

# III. Program Listings

```
C          PROGRAM ANALPOW: COMPUTES POWER SPECTRUM FROM X-Y PAIR
C       ---------------------------------------------------------------
           DIMENSION DOUT(200,2),P(200),DATA(200)
           DIMENSION XISO(200),AISO2(200),DEPTH1(200),DEPTH2(200)
           DIMENSION WORK(600)
           COMPLEX CWORK(500),WORKOUT(500),CH(5)
           CHARACTER*20 INFILE
           CHARACTER*20 OUTFILE
C       ---------------------------------------------------------------
        WRITE(6,80)
   80   FORMAT(' INPUT NUMBER OF SAMPLES IN THE DATA SEQUENCE:I3 ')
        READ(5,1000) NS
 1000   FORMAT(I3)
           WRITE(6,81)
   81      FORMAT(' WHAT IS THE INPUT FILE NAME? ')
           READ(5,82) INFILE
   82      FORMAT(A)
           OPEN(UNIT=15,NAME=INFILE,STATUS='OLD')
           WRITE(6,40)
   40      FORMAT(' WHAT IS THE OUTPUT FILE NAME ? ')
           READ(5,41) OUTFILE
   41      FORMAT(A)
           OPEN(UNIT=16,NAME=OUTFILE,STATUS='NEW')
           WRITE(6,12)
   12   FORMAT(' WHAT IS THE OUTPUT SAMPLE INTERVAL: F4.1 ? ')
        READ(5,13) XINT
   13      FORMAT(F4.1)
C       ---------------------------------------------------------------
C          INPUT DATA
           DO 1 I=1,NS
    1   READ(15,*) DEPTH1(I),XISO(I)
C          CUBIC SPLINE INTERPOLATION: NEW DEPTHS IN DEPTH2
           NPTS=((DEPTH1(NS))/XINT)+1
        DEPTH=0.0
           DEPTH2(1)=DEPTH
           DO 30 I=2,NPTS
           DEPTH =DEPTH+XINT
   30      DEPTH2(I)=DEPTH
           CALL INTERPOLATE(XISO,DEPTH1,AISO2,DEPTH2,WORK,NS,NPTS,IBAD,NF,NL)
           NOPNT=NL-NF+1
C          ANALYSIS OF DATA
  120      CALL CLEAR(CWORK,500)
C          TRANSFER DATA INTO WORK AREA
           CALL COMPLEX(AISO2,CWORK,NOPNT)
C          ZEROMEAN AND DETREND (IF NECESSARY) THE DATA
           CALL ZEROMEAN(CWORK,NOPNT)
C          WINDOW THE DATA
           CALL PROLATE(CWORK,NOPNT,1)
C          COMPUTE THE FOURIER TRANSFORM
           MOPNT=IBINARY(NOPNT)
           CALL FFT(CWORK,NOPNT,MOPNT)
           CALL POWER(CWORK,MOPNT)
C       BAND AVERAGING
        DO 20 I=1,5
   20   CH(I)=CMPLX(0.2,0.0)
        CALL CONVOLVE(CWORK,CH,WORKOUT,MOPNT,5)
        NPTS=MOPNT+4
C       CONVRT TO DB
        PEAK=-1.0E+38
```

```
      DO 50 I=1,NPTS
   50 PEAK=MAX(PEAK,REAL(WORKOUT(I)))
      DO 51 I=1,NPTS
      IF(REAL(WORKOUT(I)).EQ.0.)THEN
        WORKOUT(I)=CMPLX(-100.,0.)
        GO TO 51
      ELSE
        WORKOUT(I)=CMPLX(10.*ALOG10(REAL(WORKOUT(I))/PEAK),0.)
      END IF
   51 CONTINUE
C       OUTPUT RESULTS
        MP=(MOPNT/2)+1
        CALL REALPART(DATA,WORKOUT,NPTS)
        DO 45 I=1,(MP-1)
        AI=I
        AK=(AI/(MOPNT*XINT))
        DOUT(I,1)=AK
        IF (DATA(I+4).GT.-40.) GO TO 90
        DATA(I+4)=-40.0
   90   DOUT(I,2)=DATA(I+4)
   45   WRITE(16,46) (DOUT(I,J),J=1,2)
   46   FORMAT(1X,2G15.7)
      STOP
      END
```

```
C     PROGRAM DECOTEST: TIME-DOMAIN DECONVOLUTION WITH DECO
      DIMENSION XISO(103),H(64)
      DIMENSION XOUT(166)
      CHARACTER*20 INFILE1,INFILE3
      CHARACTER*20 OUTFILE1
C ----------------------------------------------------------------
      WRITE(6,1)
    1 FORMAT(' NUMBER OF SAMPLES IN INPUT ISOTOPE STRING?:I3 ')
      READ(5,2) NPTS
    2 FORMAT(I3)
      WRITE(6,3)
    3 FORMAT(' WHAT IS THE ISOTOPE INFILE? ')
      READ(5,4) INFILE1
    4 FORMAT(A)
      OPEN(UNIT=15,NAME=INFILE1,STATUS='OLD')
      MPTS=64
      DO 100 I=1,64
  100 H(I)=0.0
      WRITE(6,11)
   11 FORMAT(' WHAT IS THE ISOTOPE OUTFILE? ')
      READ(5,4) OUTFILE1
      OPEN(UNIT=16,NAME=OUTFILE1,STATUS='NEW')
      WRITE(6,12)
   12 FORMAT(' WHAT IS THE DECON GAIN LIMIT? :F10.9 ')
      READ(5,13) GAIN
   13 FORMAT(F10.9)
C ----------------------------------------------------------------
C     IMPULSE RESPONSE FUNCTION
      WRITE(6,50)
   50 FORMAT(' WHAT IS THE IRF INFILE? ')
      READ(5,4) INFILE3
      OPEN(UNIT=35,NAME=INFILE3,STATUS='OLD')
      DO 51 I=1,64
   51 READ(35,*) H(I)
C     READ IN DATA
      DO 22 I=1,NPTS
   22 READ(15,*) XISO(I)
      NF=NPTS+MPTS-1
      CALL DECO(H,NPTS,XISO,XOUT,GAIN)
      DO 34 I=1,NF
   34 WRITE(16,26) XOUT(I)
   26 FORMAT(G17.5)
      STOP
      END
```

```
C         PROGRAM DISSOLVE
C         PROGRAM ADJUSTS TWO STRATIGRAPHIES ABOUT INPUT CONTROL TIES
C         AND CALCULATES THE RESIDUAL.
          DIMENSION XXX1(200),XXX2(200),XXD1(200),XXD2(200)
          DIMENSION STG1(30),STG2(30),FACTOR(30),H(10)
          DIMENSION XD(200),XISO1(200),XNEW(200),XISO2(200),XOUT(200)
          DIMENSION XRESID(200),WORK(600)
          CHARACTER*20 INFILE1,INFILE2
          CHARACTER*20 OUTFILE
C         ----------------------------------------------------------------
          WRITE(6,1)
    1     FORMAT(' FILENAME FOR FIRST CORE? ')
          READ(5,2) INFILE1
    2     FORMAT(A)
          OPEN(UNIT=14,NAME=INFILE1,STATUS='OLD')
          WRITE(6,3)
    3     FORMAT(' HOW MANY SAMPLES IN THE FIRST DATA SEQUENCE:I3 ? ')
          READ(5,4) NS1
    4     FORMAT(I3)
          DO 7 I=1,NS1
    7     READ(14,99) XXD1(I),XXX1(I)
          CALL ZERO(XXX1,NS1,AMEAN1)
   99     FORMAT(5X,F5.0,3X,F5.2)
          WRITE(6,8)
    8     FORMAT(' HOW MANY STAGE BOUNDARIES ?:I2 ')
          READ(5,9) NSTG
    9     FORMAT(I2)
          WRITE(6,10)
   10     FORMAT(' INPUT STAGE BOUNDARIES IN CM ')
          DO 11 I=1,NSTG
   11     READ(5,*) STG1(I)
C         ----------------------------------------------------------
          WRITE(6,12)
   12     FORMAT(' FILENAME FOR THE SECOND CORE? ')
          READ(5,2) INFILE2
          OPEN(UNIT=15,NAME=INFILE2,STATUS='OLD')
          WRITE(6,13)
   13     FORMAT(' HOW MANY SAMPLES IN THE DATA SEQUENCE? :I3 ')
          READ(5,4) NS2
          DO 15 I=1,NS2
   15     READ(15,199) XXD2(I),XXX2(I)
          CALL ZERO(XXX2,NS2,AMEAN2)
  199     FORMAT(F4.0,F6.2)
          WRITE(6,16)
   16     FORMAT(' INPUT STAGE BOUNDARIES IN CM ')
          DO 17 I=1,NSTG
   17     READ(5,*) STG2(I)
          WRITE(6,70)
   70     FORMAT(' STAGE BOUNDARIES AND MULTIPLICATIVE FACTORS ')
          FACTOR(1)=STG1(1)/STG2(1)
          DO 71 I=2,NSTG
   71     FACTOR(I)=(STG1(I)-STG1(I-1))/(STG2(I)-STG2(I-1))
          DO 73 I=1,NSTG
   73     WRITE(6,72) I,STG1(I),STG2(I),FACTOR(I)
   72     FORMAT(I5,3F15.5)
C         ----------------------------------------------------------------
          WRITE(6,30)
   30     FORMAT(' WHAT IS THE OUTPUT FILE ? ')
          READ(5,2) OUTFILE
```

```
      OPEN(UNIT=16,NAME=OUTFILE,STATUS='NEW')
C     ------------------------------------------------------------------
C     STANDARDIZE SAMPLE SPACING IN CORE#1: 10CM INTERVALS
C     GENERATE A NEW DEPTH SCALE
      NPOINT=XXD1(NS1)/10.
      WRITE(6,49) NPOINT
   49 FORMAT(' NPOINT= ',I3)
      XD(1)=0.0
      DO 25 I=2,NPOINT
   25 XD(I)=XD(I-1)+10.0
      CALL INTERPOLATE(XXX1,XXD1,XISO1,XI,WORK,NS1,NPOINT,IBAD,NF,NI)
C     REFERENCE ISOTOPES ARE IN (XI,XISO1); SECOND SEQUENCE IS IN
C     (XXD2,XXX2). NEW DEPTH VALUES FOR CORE#2 ARE PUT IN XNEW.
      J=1
      XNEW(1)=0.0
      DO 100 I=2,NS2
      WRITE(6,42) I,XXD2(I)
      IF (XXD2(I).LE.STG2(J)) GO TO 110
      WRITE(6,43)
   43 FORMAT(' NEXT J ')
      J=J+1
  110 XNEW(I)=XNEW(I-1)+(XXD2(I)-XXD2(I-1))*FACTOR(J)
      WRITE(5,42) I,XNEW(I)
   42 FORMAT(I5,10X,F10.5)
  100 CONTINUE
C     INTERPOLATE Y VALUES FOR NEW DEPTHS
      CALL INTERPOLATE(XYX2,XNEW,XISO2,XI,WORK,NS2,NPOINT,IBAD,NF,NI)
C     CALCULATE RESIDUAL AND PLACE IN XRESID.
      DO 200 I=1,NPOINT
  200 XRESID(I)=XISO1(I)-XISO2(I)
C     SMOOTH THE RESIDUAL CURVE
      MPTS=5
      H(1)=0.11111
      H(2)=0.22222
      H(3)=0.33333
      H(4)=0.22222
      H(5)=0.11111
      CALL CNVLV(XRESID,H,XOUT,NPOINT,MPTS)
      NPTS=NPOINT+MPTS-1
      DO 201 I=1,NPTS
  201 WRITE(16,202) XD(I),XOUT(I)
  202 FORMAT(F10.1,10X,F10.5)
      STOP
      END
```

```
C PROGRAM GSTEST (GUINASSO-SCHINK TESTER)
C ------------------------------------------------------------------------
      DIMENSION A(64)
      CHARACTER*20 OUTFILE
      WRITE(6,10)
   10 FORMAT(' WHAT IS THE OUTFILE? ')
      READ(5,11) OUTFILE
   11 FORMAT(A)
      OPEN(UNIT=16,NAME=OUTFILE,STATUS='NEW')
      WRITE(6,20)
   20 FORMAT(' WHAT IS THE MIXED LAYER THICKNESS:F4.1 ')
      READ(5,21) DEP
   21 FORMAT(F4.1)
      WRITE(6,22)
   22 FORMAT(' WHAT IS THE DIFFUSION COEFFICIENT? ')
      READ(5,21) DIF
      WRITE(6,30)
   30 FORMAT(' WHAT IS THE SED RATE:F4.1 ? ')
      READ(5,31) V
   31 FORMAT(F4.1)
      CALL GSIRF(A,DEP,DIF,V)
      DO 101 I=1,64
      WRITE(16,1) A(I)
  101 WRITE(6,1) A(I)
    1 FORMAT(10X,E16.5)
      STOP
      END
```

```
C     PROGRAM HUTDEC: DECONVOLUTION WITH SIGNAL CARRIER ABUNDANCE
      DIMENSION XISO(103),POP(103),H(64)
      DIMENSION A1(103),A2(166),A3(166),A4(166),A5(166)
      CHARACTER*20 INFILE1,INFILE2,INFILE3
      CHARACTER*20 OUTFILE1
C     ------------------------------------------------------------
      WRITE(6,1)
    1 FORMAT(' NUMBER OF SAMPLES IN INPUT ISOTOPE STRING?:I3 ')
      READ(5,2) NPTS
    2 FORMAT(I3)
      WRITE(6,3)
    3 FORMAT(' WHAT IS THE ISOTOPE INFILE? ')
      READ(5,4) INFILE1
    4 FORMAT(A)
      OPEN(UNIT=15,NAME=INFILE1,STATUS='OLD')
      WRITE(6,5)
    5 FORMAT(' WHAT IS THE ABUNDANCE INFILE? ')
      READ(5,4) INFILE2
      OPEN(UNIT=25,NAME=INFILE2,STATUS='OLD')
      MPTS=64
      DO 100 I=1,64
  100 H(I)=0.0
      WRITE(6,11)
   11 FORMAT(' WHAT IS THE ISOTOPE OUTFILE? ')
      READ(5,4) OUTFILE1
      OPEN(UNIT=16,NAME=OUTFILE1,STATUS='NEW')
      WRITE(6,12)
   12 FORMAT(' WHAT IS THE DECON GAIN LIMIT? :F10.9 ')
      READ(5,13) GAIN
   13 FORMAT(F10.9)
C     ------------------------------------------------------------
C     IMPULSE RESPONSE FUNCTION
      WRITE(6,50)
   50 FORMAT(' WHAT IS THE IRF INFILE? ')
      READ(5,4) INFILE3
      OPEN(UNIT=35,NAME=INFILE3,STATUS='OLD')
      DO 51 I=1,64
   51 READ(35,*) H(I)
C     READ IN DATA
      DO 20 I=1,NPTS
   20 READ(15,21) XISO(I)
   21 FORMAT(6X,F5.2)
      DO 22 I=1,NPTS
   22 READ(25,*) POP(I)
      DO 25 I=1,NPTS
   25 A1(I)=XISO(I)*POP(I)
      NP=NPTS+MPTS-1
      CALL DECO(H,NPTS,POP,A4,GAIN)
      CALL DECO(H,NPTS,A1,A2,GAIN)
      DO 27 I=1,NP
   27 WRITE(6,26) A2(I)
      DO 28 I=1,NP
   28 WRITE(6,26) A4(I)
      DO 34 I=1,NP
      IF (A4(I).EQ.0.0) GO TO 40
      A5(I)=A2(I)/A4(I)
      GO TO 33
   40 A5(I)=0.0
   33 CONTINUE
   34 WRITE(16,26) A5(I)
   26 FORMAT(G17.5)
      STOP
      END
```

```
C       PROGRAM HUTSON: CONVOLUTION WITH SIGNAL CARRIER ABUNDANCE
        DIMENSION XISO(50),POP(50),H(64)
        DIMENSION A1(50),A2(150),A3(150),A4(150),A5(150)
        CHARACTER*20 INFILE1,INFILE2,INFILE3
        CHARACTER*20 OUTFILE1
C       ------------------------------------------------------------
        WRITE(6,1)
      1 FORMAT(' NUMBER OF SAMPLES IN INPUT ISOTOPE STRING?:I2 ')
        READ(5,2) NPTS
      2 FORMAT(I2)
        WRITE(6,3)
      3 FORMAT(' WHAT IS THE ISOTOPE INFILE? ')
        READ(5,4) INFILE1
      4 FORMAT(A)
        OPEN(UNIT=15,NAME=INFILE1,STATUS='OLD')
        WRITE(6,5)
      5 FORMAT(' WHAT IS THE ABUNDANCE INFILE? ')
        READ(5,4) INFILE2
        OPEN(UNIT=25,NAME=INFILE2,STATUS='OLD')
        MPTS=64
        DO 100 I=1,64
    100 H(I)=0.0
        WRITE(6,11)
     11 FORMAT(' WHAT IS THE ISOTOPE OUTFILE? ')
        READ(5,4) OUTFILE1
        OPEN(UNIT=16,NAME=OUTFILE1,STATUS='NEW')
C       ------------------------------------------------------------
C       IMPULSE RESPONSE FUNCTION
        WRITE(6,50)
     50 FORMAT(' WHAT IS THE IRF INFILE? ')
        READ(5,4) INFILE3
        OPEN(UNIT=35,NAME=INFILE3,STATUS='OLD')
        DO 51 I=1,64
     51 READ(35,*) H(I)
C       READ IN DATA
        DO 20 I=1,NPTS
     20 READ(15,*) XISO(I)
        DO 22 I=1,NPTS
     22 READ(25,*) POP(I)
        DO 25 I=1,NPTS
     25 A1(I)=XISO(I)*POP(I)
        NP=NPTS+MPTS-1
        CALL CNVLV(POP,H,A4,NPTS,MPTS)
        CALL CNVLV(A1,H,A2,NPTS,MPTS)
        DO 34 I=1,NP
        IF (A4(I).EQ.0.0) GO TO 40
        A5(I)=A2(I)/A4(I)
        GO TO 33
     40 A5(I)=0.0
     33 CONTINUE
     34 WRITE(16,26) A5(I)
     26 FORMAT(G17.5)
        STOP
        END
```

```
C       PROGRAM ISOJUST (ISOTOPE ADJUST)
C       ---------------------------------------------------------------------
C          PROGRAM RECALCULATES ISOTOPES WITH DEPTH/TIME ADJUSTMENTS BASED
C          ON SEDIMENTATION RATE CHANGES. DEPTH SCALE IS MODIFIED LINEARLY
C          ACCORDING TO SIGNAL MAGNITUDE.
C       ---------------------------------------------------------------------
        DIMENSION XXX(500,2),P(500),BUF(500)  ,DATA(500)
        DIMENSION XNEW(500),XN1(500),XN2(500,2),COUT(500,2),DOUT(500,2)
          DIMENSION XISO(500),XISO2(500),DEPTH1(500),DEPTH2(500)
          DIMENSION XN3(500),XN4(500),WORK(1500),STG(20),XSTG(20)
          COMPLEX CWORK(500),WORKOUT(500),CE(10)
          CHARACTER*20 INFILE
          CHARACTER*20 OUTFILE
C       ---------------------------------------------------------------------
        WRITE(6,80)
   80   FORMAT('  INPUT NUMBER OF SAMPLES IN THE DATA SEQUENCE:I3 ')
        READ(5,1000) NS
 1000   FORMAT(I3)
          WRITE (6,51)
   51     FORMAT(' WHAT IS THE INPUT FILE NAME? ')
          READ(5,41) INFILE
          OPEN(UNIT=15,NAME=INFILE,STATUS='OLD')
          WRITE(6,40)
   40     FORMAT('  WHAT IS THE OUTPUT FILE NAME ? ')
          READ(5,41) OUTFILE
   41     FORMAT(A)
          OPEN(UNIT=16,NAME=OUTFILE,STATUS='NEW')
          WRITE(6,125)
  125     FORMAT('  NUMBER OF ISOTOPE STAGES IN SERIES? :I3 ')
          READ(5,126) NISO
  126     FORMAT(I3)
          WRITE(6,127)
  127     FORMAT('  CORE DEPTHS OF ISOTOPE STAGE BOUNDARIES? :20F4.0 ')
          READ(5,128) (STG(I),I=1,NISO)
  128     FORMAT(20F4.0)
C         -------------------------------------------------------------------
C         READ DATA INTO XXX
          DO 52 I=1,NS
   52     READ(15,*) (XXX(I,J),J=1,2)
   53     FORMAT(5X,F4.0,4X,F5.2)
        NPNTS=((XXX(NS,1))/10)+1)
          WRITE(6,90) NPNTS
   90     FORMAT('  NPNTS=  ',I4)
C         INTERPOLATE DATA INTO 10CM INTERVALS
          DO 50 I=1,NS
          DEPTH1(I)=XXX(I,1)
   50     XISO(I)=XXX(I,2)
          CALL RANGE(XISO,NS,DMAX,DMIN,DX)
          WRITE(6,501),DX
  501     FORMAT(' DX=  ',G15.7)
          DO 56 I=1,NPNTS
   56     DEPTH2(I)=(I-1)*10.
          CALL INTERPOLATE(XISO,DEPTH1,XISO2,DEPTH2,CWORK,NS,NPNTS,IBAD,NF,NL
          NPTS=NL-NF+1
          WRITE(6,91) NPTS
   91     FORMAT('  NPTS=  ',I4)
C        ZEROMEAN DATA
        CALL ZERO(XISO2,NPTS,AMEAN)
C       DEPTH ADJUSTMENT
```

```
      DO 79 J=1,7,2
      FACTR=((J-1)/50.0)
      COUNT=0.0
      XNEW(1)=0.0
      DO 73 I=2,NPTS
        TERM=(2*(FACTR)*(ABS(XISO(I))))/DX
      IF(XISO2(I)) 71,71,62
   62 XNEW(I)=COUNT+(10.0)*(1.-TERM)
      COUNT=XNEW(I)
      GO TO 73
   71 XNEW(I)=COUNT+(10.0)*(1.+TERM)
      COUNT=XNEW(I)
   73 CONTINUE
        WRITE(6,150)
  150   FORMAT(' DETERMINE EFFECTIVE SEDIMENTATION DIFFERENTIALS ')
      L=1
      DO 112 I=1,NPTS
      IF (DEPTH2(I)-STG(L)) 110,111,111
  111  XSTG(L)=XNEW(I)
      L=L+1
      IF (L.GT.NISO) GO TO 750
  110  CONTINUE
  750   CONTINUE
C     WRITE OUT ADJUSTED ISOTOPE STAGE BOUNDARIES
      WRITE(6,112)
  112 FORMAT(' ADJUSTED ISOTOPE STAGE BOUNDARIES ')
      DO 113 I=1,NISO
  113  WRITE(6,114) I,XSTG(I)
  114  FORMAT(I3,10X,F4.0)
C     EFFECTIVE FACTOR = (LENGTH OF ADJUSTED STAGE)/(LENGTH OF INPUT STAGE)
      WRITE(6,116)
  116  FORMAT(' EFFECTIVE SEDIMENTATION RATE FACTORS ')
      I=1
      EFACT=((XSTG(I))/(STG(I)))
      WRITE(6,117) I,EFACT
  117  FORMAT(' ISOTOPE STAGE = ',I3,' FACTOR = ',F5.2)
      DO 118 I=2,NISO
      EFACT=((XSTG(I)-XSTG(I-1))/(STG(I)-STG(I-1)))
  118  WRITE(6,117) I,EFACT
  999   CONTINUE
C     SQUEEZE OR STRETCH XNEW TO SAME LENGTH AS XISO2.  (=NPTS)
      NP=(((XNEW(NPTS))/10)+1)
      WRITE(6,10) NP
   10 FORMAT(' NUMBER OF POINTS = ',I4)
      ANPTS=NPTS
      ANP=NP
      DO 11 I=1,NPTS
   11 XN1(I)=(XNEW(I)*(ANPTS/ANP))
C       INTERPOLATION: CUBIC SPLINE
C       XN1(I) AND XISO2(I) ARE USED AS INPUT, WITH THE NEW DEPTH SCALE
C       SPECIFIED AS XN3, OF LENGTH  NPTS .
C       RECALCULATED ISOTOPES ARE CONTAINED IN XN4.
      DEPTH=0.0
        XN3(1)=DEPTH
        DO 30 I=2,NPTS
        DEPTH =DEPTH+10.0
   30   XN3(I)=DEPTH
        CALL INTERPOLATE(XISO2,XN1,XN4,XN3,WORK,NPTS,NPTS,IBAD,NF,NL)
        NOPNT=NL-NF+1
```

```
      WRITE(6,60) NOPNT
   60 FORMAT(' NUMBER OF POINTS ARE = ',I3)
C     STORE DATA
      DO 100 I=1,NOPNT
      COUT(I,1)=(I-1)*10.
      COUT(I,2)=XN4(I)+AMEAN
  100 WRITE (16,101) (COUT(I,K),K=1,2)
  101 FORMAT(1X,2G15.7)
  888 CONTINUE
C     ANALYSIS OF DATA
      CALL CLEAR(CWORK,500)
C     TRANSFER DATA INTO WORK AREA
      CALL COMPLEX(XN4,CWORK,NOPNT)
C     ZEROMEAN AND DETREND (IF NECESSARY) THE DATA
      CALL ZEROMEAN(CWORK,NOPNT)
C     WINDOW THE DATA
      CALL PROLATE(CWORK,NOPNT,0)
C     COMPUTE THE FOURIER TRANSFORM
      MOPNT=IBINARY(NOPNT)
      CALL FFT(CWORK,NOPNT,MOPNT)
      CALL POWER(CWORK,MOPNT)
C     BAND AVERAGING
      CH(1)=CMPLX(0.25,0.)
      CH(2)=CMPLX(0.5,0.)
      CH(3)=CMPLX(0.25,0.)
      CALL CONVOLVE(CWORK,CH,WORKOUT,MOPNT,3)
C      CONVERT TO RELATIVE DECIBELS
      PEAK=-1.0E+38
      DO 190 I=1,MOPNT+2
  190 PEAK=MAX(PEAK,REAL(WORKOUT(I)))
      DO 191 I=1,MOPNT+2
      IF (REAL(WORKOUT(I)).EQ.0.)THEN
         WORKOUT(I)=CMPLX(-100.,0.)
         GO TO 191
      ELSE
         WORKOUT(I)=CMPLX(10.*ALOG10(REAL(WORKOUT(I))/PEAK),0.)
      END IF
  191 CONTINUE
C      OUTPUT RESULTS
       MP=(MOPNT/2)
       CALL REALPART(DATA,WORKOUT,MOPNT+2)
       DO 45 I=2,MP+1
      AI=I-1
      AK=AI/(MOPNT*10.)
       IF (DATA(I+1).GT.-25.) GO TO 45
       DATA(I+1)=-25.
   45 WRITE(16,47) AK,DATA(I+1)
   47 FORMAT(5X,G15.7,5X,G17.5)
C     VARIMAX CALCULATION
      V1=0.0
      V2=0.0
      V=0.0
      DO 95 I=3,MP+2
      V1=V1+(DATA(I)**4)
      V2=V2+(DATA(I)**2)
   95 CONTINUE
      V=V1/(V2**2)
      WRITE(6,46) J-1,V
   46 FORMAT(5X,I2,5X,G15.7)
```

```
   79 CONTINUE
  610 CONTINUE
      STOP
      END
      SUBROUTINE RANGE(X,NS,XMAX,XMIN,DX)
C     PROGRAM DETERMINES MAX,MIN VALUES AND AMPLITUDE OF A VECTOR.
      DIMENSION X(500)
      XMIN=X(1)
      XMAX=XMIN
      DO 50 I=1,NS
      IF(X(I).LT.XMIN) XMIN=X(I)
      IF(X(I).GT.XMAX) XMAX=X(I)
   50 CONTINUE
      DX=XMAX-XMIN
      RETURN
      END
```

```
C      PROGRAM OFFSET: SIMPLEX DECONVOLUTION
C      SIMPLEX OPTIMIZATION OF DANIELS,1978, MODIFIED
C      TO OPTIMIZE THE FIT OF TWO UNMIXED ISOTOPE SIGNALS.
       DIMENSION C(2),ERR(3),P(3,3),R(2),X(2)
       DIMENSION XISO1(100),XISO2(100),POP1(100),POP2(100),W(100)
       DIMENSION PW1(100),PW2(100),PWI1(100),PWI2(100),OUT1(128)
       DIMENSION OUT3(128)
       INTEGER H
       CHARACTER*20 INFILE1,INFILE2,INFILE3,INFILE4,INFILE5
       CHARACTER*20 OUTFILE
C      ---------------------------------------------------------------
       WRITE(6,200)
   200 FORMAT(' HOW MANY DATA POINTS:I2 ? ')
       READ(5,201) NP
   201 FORMAT(I2)
C      READ IN DATA FILES
       WRITE(6,202)
   202 FORMAT(' ISO1 FILENAME ? ')
       READ(5,203) INFILE1
   203 FORMAT(A)
       OPEN(UNIT=15,NAME=INFILE1,STATUS='OLD')
       DO 204 I=1,NP
   204 READ(15,205) XISO1(I)
       CALL ZERO(XISO1,NP,AMEAN1)
       DO 888 I=1,NP
   888 XISO1(I)=XISO1(I)*0.87
   205 FORMAT(F6.2)
       WRITE(6,206)
   206 FORMAT(' ISO2 INFILE? ')
       READ(5,203) INFILE2
       OPEN(UNIT=17,NAME=INFILE2,STATUS='OLD')
       DO 207 I=1,NP
   207 READ(17,205) XISO2(I)
       CALL ZERO(XISO2,NP,AMEAN2)
       WRITE(6,208)
   208 FORMAT(' POP1 FILENAME ? ')
       READ(5,203) INFILE3
       OPEN(UNIT=18,NAME=INFILE3,STATUS='OLD')
       DO 209 I=1,NP
   209 READ(18,205) POP1(I)
       WRITE(6,210)
   210 FORMAT(' POP2 FILENAME ? ')
       READ(5,203) INFILE4
       OPEN(UNIT=19,NAME=INFILE4,STATUS='OLD')
       DO 211 I=1,NP
   211 READ(19,205) POP2(I)
       WRITE(6,212)
   212 FORMAT(' WEIGHT FILENAME ? ')
       READ(5,203) INFILE5
       OPEN(UNIT=20,NAME=INFILE5,STATUS='OLD')
       DO 213 I=1,NP
   213 READ(20,205) W(I)
C      ---------------------------------------------------------------
C      CALCULATE POPULATION CHANGES
       DO 220 I=1,NP
       PW1(I)=POP1(I)*W(I)
       PW2(I)=POP2(I)*W(I)
       PWI1(I)=PW1(I)*XISO1(I)
   220 PWI2(I)=PW2(I)*XISO2(I)
```

```
C          ------------------------------------------------------------
C          READ IN INITIAL VALUES
           WRITE(6,10)
        10 FORMAT(' WHAT ARE THE INITIAL VALUES:DEP,DIF:2F5.2? ')
           READ(5,11) X(1),X(2)
        11 FORMAT(2F5.2)
           WRITE(6,15)
        15 FORMAT(' WHAT IS THE SEI RATE:F4.1 ? ')
           READ(5,16) V
        16 FORMAT(F4.1)
           WRITE(6,13)
        13 FORMAT(' WHAT IS THE DECON GAIN LIMITATION:F10.9 ? ')
           READ(5,14) GAIN
        14 FORMAT(F10.9)
           CALL ERROR(X,E,V,PWI1,PWI2,PW1,PW2,GAIN,NP,OUT1,OUT3)
           WRITE(6,12) X(1),X(2),E
        12 FORMAT(' DEP= ',F10.5,5X,' DIF= ',F10.5,5X,' E= ',F10.5)
C          ------------------------------------------------------------
C          INITIALIZE SIMPLEX
           DO 22 J=1,2
        22 P(1,J)=X(J)
           DO 28 I=2,3
           DO 26 J=1,2
        26 P(I,J)=X(J)
           P(I,I-1)=1.1*X(I-1)
        28 IF(ABS(X(I-1)).LT.1E-12) P(I,I-1)=0.0001
C          ORDER POINTS ACCORDING TO VALUES OF ERROR
        31 L=1
           H=1
           DO 38 I=1,3
           DO 35 J=1,2
        35 X(J)=P(I,J)
           CALL ERROR(X,E,V,PWI1,PWI2,PW1,PW2,GAIN,NP,OUT1,OUT3)
           ERR(I)=E
           IF(ERR(I).LT.ERR(L)) L=I
        38 IF(ERR(I).GT.ERR(H)) H=I
        41 NH=L
           DO 43 I=1,3
        43 IF(ERR(I).GE.ERR(NH).AND.I.NE.H) NH=I
C          ------------------------------------------------------------
C          CALCULATE CENTROID (AVERAGE)
           DO 56 J=1,2
           C(J)=-P(H,J)
           DO 54 I=1,3
        54 C(J)=C(J)+P(I,J)
        56 C(J)=C(J)/2.
           WRITE(6,502)
       502 FORMAT(' REFLECT ABOUT THE CENTROID')
        61 DO 62 J=1,2
        62 R(J)=1.9985*C(J)-0.9985*P(H,J)
           CALL ERROR(R,ER,V,PWI1,PWI2,PW1,PW2,GAIN,NP,OUT1,OUT3)
           WRITE(6,12) R(1),R(2),ER
C          REFLECT AGAIN (IF MODERATELY SUCCESSFUL)
           IF(ER.LT.ERR(L)) GO TO 91
           IF(ER.GE.ERR(H)) GO TO 122
        79 DO 80 J=1,2
        80 P(H,J)=R(J)
           ERR(H)=ER
           IF(ER.GT.ERR(NH)) GO TO 61
```

```
      H=NH
      GO TO 41
C     ------------------------------------------------------------------
   91 L=H
      WRITE(6,503)
  503 FORMAT(' EXPAND THE SIMPLEX ')
      H=NH
      DO 93 J=1,2
   93 X(J)=1.95*R(J)-0.95*C(J)
      CALL ERROR(X,EX,V,FWI1,FWI2,PW1,PW2,GAIN,NP,OUT1,OUT3)
      WRITE(6,12) X(1),X(2),EX
      IF(EX.LT.ER) GO TO 104
      DO 99 J=1,2
   99 P(L,J)=R(J)
      ERR(L)=ER
      GO TO 110
  104 DO 105 J=1,2
  105 P(L,J)=X(J)
      ERR(L)=EX
  110 WRITE(6,12) P(L,1),P(L,2),ERR(L)
      WRITE(6,111)
  111 FORMAT(' DO YOU WISH TO CONTINUE? YES=1,NO=0 ')
      READ(5,112) IFI
  112 FORMAT(I1)
      IF(IFI.EQ.1) GO TO 41
      GO TO 150
C     ------------------------------------------------------------------
  122 WRITE(6,504)
  504 FORMAT(' CONTRACT THE SIMPLEX ')
      DO 123 J=1,2
  123 R(J)=0.5015*C(J)+0.4985*P(H,J)
      CALL ERROR(R,ER,V,FWI1,FWI2,PW1,PW2,GAIN,NP,OUT1,OUT3)
      WRITE(6,12) R(1),R(2),ER
      IF(ER.LT.ERR(L)) GO TO 91
      IF(ER.LT.ERR(H)) GO TO 79
C     SCALE THE SIMPLEX: SK IS THE SCALING FUNCTION
      WRITE(6,130)
  130 FORMAT(' WHAT IS THE SIMPLEX SCALING FACTOR:F4.1 ? ')
      READ(5,131) SK
  131 FORMAT(F4.1)
      DO 138 I=1,3
      DO 138 J=1,2
  138 P(I,J)=P(I,J)+SK*(P(I,J)-P(I,J))
      GO TO 31
  150 CONTINUE
      WRITE(6,151)
  151 FORMAT(' WHAT IS THE OUTFILE ? ')
      READ(5,203) OUTFILE
      OPEN(UNIT=16,NAME=OUTFILE,STATUS='NEW')
      NPTS=NP+64-1
      A=(NPTS/2)-20
      B=(NPTS/2)+20
      DO 152 I=A,B
      OUT1(I)=OUT1(I)*1.15
      WRITE(16,153) I,OUT1(I)+AMEAN1,OUT3(I)+AMEAN2
  152 WRITE(6,153) I,OUT1(I)+AMEAN1,OUT3(I)+AMEAN2
  153 FORMAT(I5,10X,G17.5,10X,G17.5)
      STOP
      END
```

```
C PROGRAM PROJECT: COMPUTES POWER SPECTRUM IN DB
      DIMENSION XIN(128),XOUT(128)
      COMPLEX WORK(128)
      CHARACTER*20 INFILE,OUTFILE
C     ------------------------------------------------------------
      WRITE(6,1)
    1 FORMAT(' NUMBER OF SAMPLES IN INPUT :I2 ? ')
      READ(5,2) NPTS
    2 FORMAT(I2)
      WRITE(6,3)
    3 FORMAT(' WHAT IS THE INFILE? ')
      READ(5,4) INFILE
    4 FORMAT(A)
      OPEN(UNIT=15,NAME=INFILE,STATUS='OLD')
      WRITE(6,5)
    5 FORMAT(' WHAT IS THE OUTFILE? ')
      READ(5,4) OUTFILE
      OPEN(UNIT=16,NAME=OUTFILE,STATUS='NEW')
      WRITE(6,6)
    6 FORMAT(' WHAT IS THE SED RATE:F5.2 ? ')
      READ(5,7) V
    7 FORMAT(F5.2)
C     ------------------------------------------------------------
C     READ IN DATA
      DO 10 I=1,NPTS
   10 READ(15,11) XIN(I)
   11 FORMAT(25X,F6.5)
      CALL CLEAR(WORK,128)
      CALL COMPLEX(XIN,WORK,128)
      CALL FFT(WORK,NPTS,128)
      CALL POWERDB(WORK,128)
      CALL REALPART(XOUT,WORK,128)
      CINT=NPTS/V
      DO 100 I=2,65
      XI=I-1
      FREQ=XI/CINT
  100 WRITE(16,122) FREQ,XOUT(I)
  122 FORMAT(F10.5,10X,F10.5)
      STOP
      END
```

```
C      PROGRAM VARIMAX: SIMPLICITY NORM
       DIMENSION XIN(128)
       CHARACTER*20 INFILE
C      ------------------------------------------------
       WRITE(6,1)
     1 FORMAT(' HOW MANY SAMPLES IN THE INPUT ? :I2 ')
       READ(5,2) N
     2 FORMAT(I2)
       WRITE(6,3)
     3 FORMAT(' WHAT IS THE INFILE ? ')
       READ(5,4) INFILE
     4 FORMAT(A)
       OPEN(UNIT=15,NAME=INFILE,STATUS='OLD')
       DO 9 I=1,N
     9 READ(15,*) XIN(I)
       V1=0.0
       V2=0.0
       V=0.0
       DO 11 I=1,N
       V1=V1+(XIN(I)**4)
    11 V2=V2+(XIN(I)**2)
       V=V1/(V2**2)
       WRITE(6,100) V
   100 FORMAT(' THE VARIMAX NORM IS ',G15.7)
       STOP
       END
```

```
C      PROGRAM WAVETEST
C      ------------------------------------------------------------
C      PROGRAM GENERATES A MULTI-WAVELENGTH INPUT SIGNAL, AND
C      SIMULATES SEDIMENTATION RATE CHANGES WITH AN AUTO-MODULATION
C      BASED ON SIGNAL AMPLITUDE. THIS PROGRAM IS SIMILAR TO PROGRAM
C      ISOJUST, WHICH ACCEPTS OXYGEN ISOTOPE DATA AS THE CONTROLLING
C         INPUT.
C      ------------------------------------------------------------
       DIMENSION XXX(500,2),P(500),BUF(500)   ,DATA(500)
       DIMENSION XNEW(500),XN1(500),XN2(500,2),XISO(500)
          DIMENSION XN3(500),XN4(500),WORK(1500)
          COMPLEX CWORK(500)
          CHARACTER*20 OUTFILE
C      ------------------------------------------------------------
       WRITE(6,80)
    80 FORMAT('  INPUT NUMBER OF SAMPLES IN THE DATA SEQUENCE:I3 ')
       READ(5,1000) NS
  1000 FORMAT(I3)
          WRITE(6,40)
    40    FORMAT(' WHAT IS THE OUTPUT FILE NAME ? ')
          READ(5,41) OUTFILE
    41    FORMAT(A)
          OPEN(UNIT=16,NAME=OUTFILE,STATUS='NEW')
C      ------------------------------------------------------------
C      GENERATE RAW DATA
       CALL WAVGEN(XISO,NS)
       CALL RANGE(XISO,NS,DMAX,DMIN,DX)
       CALL MEAN(XISO,NS,DMEAN)
       WRITE(6,15) DMAX,DMIN,DX,DMEAN
    15 FORMAT('  MAX,MIN,RANGE, AND MEAN   VALUE OF ISOTOPE CURVE: ',
     14(2X,F8.2))
C         LOAD XISO INTO XXX
          DO 50 I=1,NS
          XXX(I,1)=(I-1)*10.0
    50    XXX(I,2)=XISO(I)
       NPTS=((XXX(NS,1))/10)+1
          WRITE(6,90) NPTS
    90    FORMAT('  NPTS= ',I4)
C      DEPTH ADJUSTMENT
       DO 79 J=1,6
       FACTR=((J-1)/10.0)
       COUNT=0.0
       XNEW(1)=0.0
       DO 73 I=2,NS
          TERM=(2*(FACTR)*(ABS(XXX(I,2))))/DX
       IF(XXX(I,2)) 62,71,71
    62 XNEW(I)=COUNT+(10.0)*(1.-TERM)
       COUNT=XNEW(I)
       GO TO 73
    71 XNEW(I)=COUNT+(10.0)*(1.+TERM)
       COUNT=XNEW(I)
    73 CONTINUE
C      SQUEEZE OR STRETCH XNEW TO SAME LENGTH AS XXX.  (=NPTS)
       NP=(((XNEW(NS)))/10)+1
       WRITE(6,10) NP
    10 FORMAT('  NUMBER OF POINTS = ',I4)
       ANPTS=NPTS
       ANP=NP
       DO 11 I=1,NS
```

```
   11 XN1(I)=(XNEW(I)*(ANPTS/ANP))
C        INTERPOLATION: CUBIC SPLINE
C        XN1(I) AND XISO(I) ARE USED AS INPUT, WITH THE NEW DEPTH SCALE
C        SPECIFIED AS XN3, OF LENGTH  NPTS .
C        RECALCULATED ISOTOPES ARE CONTAINED IN XN4.
      DEPTH=0.0
         XN3(1)=DEPTH
         DO 30 I=2,NPTS
         DEPTH =DEPTH+10.0
   30    XN3(I)=DEPTH
         CALL INTERPOLATE(XISO,XN1,XN4,XN3,WORK,NS,NPTS,IBAL,NF,NL)
         NOPNT=NL-NF+1
         WRITE(6,60) NOPNT
   60    FORMAT(' NUMBER OF POINTS ARE = ',I3)
C        STORE DATA
         DO 100 I=1,NOPNT
  100    WRITE (16,101) XN4(I)
  101    FORMAT(1X,G15.7)
C        ANALYSIS OF DATA
         CALL CLEAR(CWORK,500)
C        TRANSFER DATA INTO WORK AREA
         CALL COMPLEX(XN4,CWORK,NOPNT)
C        ZEROMEAN AND DETREND (IF NECESSARY) THE DATA
         CALL ZEROMEAN(CWORK,NOPNT)
C        WINDOW THE DATA
         CALL PROLATE(CWORK,NOPNT,0)
C        COMPUTE THE FOURIER TRANSFORM
         MOPNT=IBINARY(NOPNT)
         CALL FFT(CWORK,NOPNT,MOPNT)
         CALL POWERDE(CWORK,MOPNT)
C        OUTPUT RESULTS
         MP=(MOPNT/2)
         CALL REALPART(DATA,CWORK,MOPNT)
         DO 45 I=1,MP
         XI=I
      FREQ=XI/MOPNT
         IF (DATA(I+1).GT.-40.) GO TO 45
         DATA(I+1)=-40.
   45    WRITE(15,46) FREQ,DATA(I+1)
   46    FORMAT(1X,G15.7,10X,G15.7)
   79 CONTINUE
      STOP
      END
      SUBROUTINE RANGE(X,NS,XMAX,XMIN,IX)
C     PROGRAM DETERMINES MAX,MIN VALUES AND AMPLITUDE OF A VECTOR.
      DIMENSION X(500)
      XMIN=X(1)
      XMAX=XMIN
      DO 50 I=1,NS
      IF(X(I).LT.XMIN) XMIN=X(I)
      IF(X(I).GT.XMAX) XMAX=X(I)
   50 CONTINUE
      DX=XMAX-XMIN
      RETURN
      END
      SUBROUTINE MEAN(X,NS,AVG)
C     SUBROUTINE DETERMINES MEAN VALUE OF A VECTOR
      DIMENSION X(500)
      SUM=0.0
```

```
      DO 50 I=1,NS
   50 SUM=SUM+X(I)
      AVG=(SUM)/(NS)
      RETURN
      END
```

```
      SUBROUTINE CNVLV(A,B,C,N,M)
C     CONVOLVES A(N) WITH B(M).RESULT IS STORED IN C(N+M-1).
      DIMENSION A(1),B(1),C(1)
      DO 5 I=1,N+M-1
    5 C(I)=0.0
      DO 10 I=1,N
      DO 10 J=1,M
   10 C(I+J-1)=C(I+J-1)+A(I)*B(J)
      RETURN
      END
```

```
      SUBROUTINE DECO(H,NP,XIN,XOUT,CNST)
C     SUBROUTINE DECO: TIME-DOMAIN DECONVOLUTION
      DIMENSION H(64),H1(64)
      DIMENSION XIN(100),XOUT(128)
      COMPLEX WORK(128)
C     ------------------------------------------------------------
C     FOURIER TRANSFORM OF IRF
      CALL CLEAR(WORK,128)
      CALL COMPLEX(H,WORK,64)
      CALL FFT(WORK,64,64)
C     BRING UP MODULUS WITH GAIN LIMITER
      CALL POLAR(WORK,64)
      CALL REALPART(H1,WORK,64)
      DO 85 I=1,64
      H1(I)=H1(I)+CNST
   85 WORK(I)=CMPLX(H1(I),AIMAG(WORK(I)))
      CALL CARTESIAN(WORK,64)
      DO 110 I=1,64
  110 WORK(I)=(1.0,0.0)/WORK(I)
      CALL FFTINVERSE(WORK,64)
      CALL REALPART(H1,WORK,64)
C     NORMALIZE COEFFICIENTS TO UNITY
      SUM=0.0
      DO 125 I=1,64
  125 SUM=SUM+H1(I)
      DO 126 I=1,64
  126 H1(I)=H1(I)/SUM
      CALL CNVLV(XIN,H1,XOUT,NP,64)
      RETURN
      END
```

```
      SUBROUTINE ERROR(X,E,V,PWI1,PWI2,PW1,PW2,GAIN,NP,OUT1,OUT3)
      DIMENSION H(64),OUT1(128),OUT2(128),OUT3(128),OUT4(128)
      DIMENSION X(1),PWI1(1),PWI2(1),PW1(1),PW2(1)
C     ---------------------------------------------------------------
C     GENERATE IRF
      E=Ø.Ø
      DEP=X(1)
      DIF=X(2)
      CALL GSIRF(H,DEP,DIF,V)
      CALL DECO(H,NP,PW1,OUT1,GAIN)
      CALL DECO(H,NP,PWI1,OUT2,GAIN)
      CALL DECO(H,NP,PW2,OUT3,GAIN)
      CALL DECO(H,NP,PWI2,OUT4,GAIN)
C     EXAMINE ERROR OVER CENTER POINTS
      NPTS=NP+63
      A=(NPTS/2)-2Ø
      B=(NPTS/2)+2Ø
      DO 1Ø I=A,B
      OUT1(I)=OUT2(I)/OUT1(I)
   1Ø OUT3(I)=OUT4(I)/OUT3(I)
      A1=(NPTS/2)
      B1=(NPTS/2)+9
      DO 2Ø I=A1,B1
      TERM=ABS(OUT1(I)-OUT3(I))
   2Ø E=E+TERM
      RETURN
      END
```

```
      SUBROUTINE GSIRF(AA,DEP,DIF,V)
C     SUBROUTINE USES OFFICER AND LYNCH(1983) APPROXIMATION OF THE
C     GUINASSO-SCHINK EQUATION TO GENERATE AN IMPULSE RESPONSE FUNCTION
C     IRF IS 64 POINTS LONG
      DIMENSION AA(64)
C     PARAMETER SETUP: V=SED RATE, XI0=TRACER IMPACT DEPTH
      XI0=40.0
      XTOT=0.0
      WRITE(6,100) DEP,DIF
  100 FORMAT(' DEP= ',F10.5,5X,' DIF= ',F10.5)
      PI=3.14159
      DO 10 I=1,64
      IDEX=I
      XI=I
      A1=(V/(4.*DIF))*(XI+DEP-XI0)
      B1=-(XI0+DEP-XI)/DEP
      C1=-((XI0+DEP-XI)/V)*(PI*PI*DIF)/(DEP**2)
      D1=-((XI0+DEP-XI)/V)*(4*PI*PI*DIF)/(DEP**2)
      E1=-((XI0+DEP-XI)/V)*(9*PI*PI*DIF)/(DEP**2)
      A=EXP(A1)
      B=(1./DEP)*EXP(B1)
      C=(2./DEP)*EXP(C1)
      D=(2./DEP)*EXP(D1)
      E=(2./DEP)*EXP(E1)
      AA(I)=A*(B-C+D-E)
   10 IF (AA(I).LT.0.0) GO TO 20
   20 DO 21 J=IDEX,64
   21 AA(J)=0.0
      DO 25 I=1,64
   25 XTOT=XTOT+AA(I)
      WRITE(6,22) XTOT
   22 FORMAT(' XTOT= ',F10.5)
      DO 30 I=1,64
   30 AA(I)=AA(I)/XTOT
      RETURN
      END
```

```
      SUBROUTINE WAVGEN(X,NS)
C     ****************************************************************
C     INSERT: WAVGEN GENERATES A MULTI-WAVELENGTH SIGNAL
C     WAVE AMPLITUDES ARE SCALARS:AMP,BMP,CMP,DMP
C     WAVE FREQUENCIES ARE HELD AS (2*PI)/(WAVELENGTH): AF,BF,CF,DF
      DIMENSION A(500),B(500),C(500),D(500),X(500)
      PI=3.14159
      AMP=1.0
      BMP=1.0
      CMP=1.0
      DMP=0.0
      AF=(2*PI)/21.0
      BF=(2*PI)/(8.22)
      CF=(2*PI)/(4.62)
      DF=(2*PI)/(23.)
      WRITE(6,1000)
 1000 FORMAT(' COMPONENTS OF THE INPUT TEST SIGNAL ')
      DO 10 I=1,NS
   10 A(I)=AMP*(SIN(AF*I))
      WRITE(6,1001) AMP,AF
 1001 FORMAT('      WAVE  AMPLITUDE = ',F6.2,'; 2*PI/WVLGTH = ',F6.2)
      DO 20 I=1,NS
   20 B(I)=BMP*(SIN(BF*I))
      WRITE(6,1001) BMP,BF
      DO 30 I=1,NS
   30 C(I)=CMP*(SIN(CF*I))
      WRITE(6,1001) CMP,CF
      DO 40 I=1,NS
   40 D(I)=DMP*(SIN(DF*I))
      WRITE(6,1001) DMP,DF
      DO 100 I=1,NS
  100 X(I)=A(I)+B(I)+C(I)+D(I)
      RETURN
      END
```

```
C     ZEROMEAN ALGORITHM
      SUBROUTINE ZERO(A,N,AMEAN)
      DIMENSION A(1)
      AMEAN=0.0
      DO 10 I=1,N
   10 AMEAN=AMEAN+A(I)/N
      DO 20 I=1,N
   20 A(I)=A(I)-AMEAN
      RETURN
      END
```

## IV. References


Adelseck, C., 1980. Modification of Pleistocene carbonate preservation by a 500,000 yr. dissolution cycle. Int. Geol. Cong. Abstr., 26: 418.

Berger, W. H., and Killingley, J. S., 1982. Box cores from the equatorial Pacific: $^{14}C$ sedimentation rates and benthic mixing. Mar. Geol., 45: 93-125.

Berger, W. H., Johnson, R. F., and Killingley, J. S., 1977. "Unmixing" of the deep-sea record and the deglacial meltwater spike. Nature, 269: 661-663.

Chave, A. D., 1980. Time series analysis programs. Woods Hole Oceanographic Institution Technical Report, WHOI-80-23, 53 p.

Daniels, R. W., 1978. An introduction to numerical methods and optimization techniques. North-Holland, New York.

de Boor, C., 1978. A practical guide to splines. Springer-Verlag, Berlin.

Guinasso, N. L., and Schink, D. R., 1975. Quantitative estimates of biological mixing in abyssal sediments. J. Geophys. Res., 80: 3032-3043.

Hutson, W. H., 1980. Bioturbation of deep-sea sediments: oxygen isotopes and stratigraphic uncertainty. Geology, 8: 127-130.

Nelder, J. A., and Mead, R., 1965. A simplex method for function minimization. Comput. J., 7: 308-313.

Officer, C. B., and Lynch, D. R., 1983. Determination of mixing parameters from tracer distributions in deep-sea sediment cores. Mar. Geol., 52: 59-74.

Otnes, R. K., and Enochson, L., 1972. Digital time series analysis. Wiley, New York.

Schiffelbein, P. A., 1984. Stable isotope systematics in Pleistocene deep-sea sediment records. Ph.D. Dissertation, Scripps Institution of Oceanography.

Wiggins, R. A., 1978. Minimum entropy deconvolution. Geoexploration, 16: 21-35.

END

FILMED

1 84

DTIC